2011 50th IEEE Conference on Decision and Control and
European Control Conference (CDC-ECC)
Orlando, FL, USA, December 12-15, 2011

Total Unimodularity and Degeneracy-Aware Dantzig-Wolfe Decomposition for Large-Capacity Cell Transmission Model

Peng Wei and Dengfeng Sun
School of Aeronautics and Astronautics
Purdue University, West Lafayette, IN 47907, USA
Email: {weip, dsun}@purdue.edu

***Abstract*—In an earlier work, Sun and Bayen built a Large-Capacity Cell Transmission Model for air traffic flow management. They formulated an integer programming problem of minimizing the total travel time of flights in the National Airspace System of the United States subject to sector capacity constraints. The integer programming was relaxed to a linear programming for computational efficiency. In this paper the authors formulate the optimization problem in a standard linear programming form. We analyze the total unimodular property of the constraint matrix, and prove that the linear programming relaxation generates an integral optimal solution for the original integer programming. It is guaranteed to be optimal and integral if solved by the simplex method. Furthermore, we find the degeneracies for both feasible polyhedron and optimal polyhedron. In order to speed up the computation, we present a revised Dantzig-Wolfe Decomposition algorithm, which is shown to preserve the total unimodularity of the constraint matrix and successfully resolve the degeneracies.**

I. INTRODUCTION

Optimization techniques have been developed to facilitate *Traffic Flow Management* (TFM). Current popular TFM schemes mainly focus on ground delay and flight rerouting. Odoni was the first to formulate the TFM problem to design optimal strategies by assigning ground delays to flights [1]. Helme was among the first who included en route capacity restrictions [2]. Lindsay proposed a disaggregate 0-1 programming model for deciding ground and airborne holding under airport and airspace capacity constraints [3]. Sridhar presented an integrated three-step hierarchical method for determining TFM plans consisting of detailed practical restrictions [4].

In [5] the authors presented a traffic flow model called the *Large-capacity Cell Transmission Model*, in short CTM(L), which is a variation of the model in [6]. They applied it to a problem of minimizing the total travel time of all flights in the NAS of the United States restricted by sector capacity counts, which is an integer programming containing billions of variables and constraints. It was then relaxed to a *linear programming* (LP) for computational efficiency. However, solving the linear programming by large scale commercial software can possibly result in fractional optimal solutions which can not be implemented as en route holding control in practice directly. This is the major motivation of our work.

In this paper we study the solution space structure of the problem and prove that there exists an integral optimal solution in the linear programming relaxation, which is also optimal for the original integer programming. The solution is guaranteed to be integral when solved by the simplex method. Moreover, we find the solution space degeneracies inside the feasible polyhedron and the optimal polyhedron of the problem. Therefore we propose the simplex based Degeneracy-Aware Dantzig-Wolfe Decomposition to ensure integral optimum, while achieving a faster computation speed and sustaining the degeneracies.

The rest of this paper is organized as follows. The second section introduces the CTM(L) model. The third section formulates the integer programming problem in a standard linear programming form and analyzes its total unimodularity. In the fourth section, a small scale model is demonstrated and the trade-off among different algorithms is discussed. In Sec. V we develop the Degeneracy-Aware Dantzig-Wolfe Decomposition algorithm. Sec. VI concludes the paper.

II. CTM(L) AND ITS MATHEMATICAL FORMULATION

A. Construction of the network

The network flow model is composed of nodes and links. The nodes are created as the entry and exit points at the sector boundaries. For any sectors s_1, s_2 and s_3, if s_1 and s_2 share a boundary and if s_2 and s_3 are neighbors, two *directed links* are created: one from node $v_{\{s_1,s_2\}}$ to node $v_{\{s_2,s_3\}}$ and one from node $v_{\{s_3,s_2\}}$ to node $v_{\{s_2,s_1\}}$.

The expected travel time of a flight through a link is computed from ASDI/ETMS data [7], which is used to determine the length of the link. Each link is divided into several *cells* as time interval units.

B. Dynamics

The CTM(L) model inspired by [6] and [8] is reduced to a linear time-invariant dynamical system.

The air traffic flow on link i can be depicted as *Link Level Model* [5]:

$$\begin{array}{l} x_i(t+1) = A_i x_i(t) + B_1^i u_i(t) + B_2^i f_i(t), \\ y(t) = \tilde{C}_i' x_i(t) \end{array} \qquad (1)$$

where $x_i(t) = [x_i^1(t), ..., x_i^{N_i}(t)]$ is the state vector whose elements represent the corresponding aircraft counts in each cell of link i at time t, and N_i is the number of cells in link

i. The forcing scalar input $f_i(t)$ denotes the entry count onto link i during time t, and the vector $u_i(t)$ represents holding control. The output $y(t)$ is the aircraft count in a user-specified set of cells at time t. $\tilde{C}_i$ is the user-specified index vector. A_i is an $N_i \times N_i$ nilpotent matrix with 1's on its super-diagonal. $B_2^i = [1;0;...;0]$ is the forcing vector with N_i elements, and B_1^i is the $N_i \times N_i$ holding pattern matrix, in which all non-zero elements are 1 on the diagonal and -1 on the super-diagonal.

Based on the link level model, it is easy to build a *Sector Level Model* by integrating all the links in a sector, e.g. the matrices A_i of different links in Eq. (1) are put in diagonal blocked matrix A and the vectors x_i are cascaded as x.

The NAS-wide model can also be cast in the same procedure. Further details about the CTM(L) are described in [5].

III. Problem Formulation and Totally Unimodular Property

A. Integer Programming Formulation

1) Link Level Model: According to [5], for a single path with N cells, the initial condition of the model is:

$$x_k(0) = x_k^0, \quad k = 0, 1, ..., N-1 \tag{2}$$

the boundary conditions are:

$$\begin{aligned} &x_0(0) = f(0) + x_0^0, \\ &x_0(t) = f(t) + u_0(t-1), \quad t = 1, 2, ..., T-1 \end{aligned} \tag{3}$$

and the dynamics are:

$$\begin{aligned} &x_k(t) = x_{k-1}(t-1) - u_{k-1}(t-1) + u_k(t-1), \\ &k = 1, 2, ..., N-1, \quad t = 1, 2, ..., T-1 \end{aligned} \tag{4}$$

where T is the planning horizon.

We cascade all the $x_k(t)$ into a vector x in the sequence as below:

$$\begin{aligned} x = [&x_0(0), ..., x_{N-1}(0), x_0(1), ..., x_{N-1}(1), \\ &..., x_0(T-1), ..., x_{N-1}(T-1)]'. \end{aligned} \tag{5}$$

Similarly, vector u is created of the same length NT as x:

$$u = [u_0(0), ..., u_{N-1}(T-1)]'. \tag{6}$$

We integrate both initial states (2) and boundary states (3) into a vector f of length NT:

$$\begin{aligned} f = [&f(0) + x_0(0), x_1(0), ..., x_{N-1}(0), \\ &f(1), 0, ..., 0, ..., f(T-1), 0, ..., 0]'. \end{aligned} \tag{7}$$

Finally, an equality form is generated by combining Eqs. (5) (6) (7):

$$x = Px + Qu + f \tag{8}$$

where matrices P and Q are both of dimension $NT \times NT$.

Matrix P is:

$$P_{(NT\times NT)} = \begin{pmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ P_o & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & P_o & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & P_o & \mathbf{0} \end{pmatrix}, \tag{9}$$

where $\mathbf{0}$ and P_o are both $N \times N$ matrices. Matrix P_o is:

$$P_{o(N\times N)} = \begin{pmatrix} 0 & 0 & 0 & \dots & 0 & 0 \\ 1 & 0 & 0 & \dots & 0 & 0 \\ 0 & 1 & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & \dots & 1 & 0 \end{pmatrix}. \tag{10}$$

Matrix Q is:

$$Q_{(NT\times NT)} = \begin{pmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ Q_o & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & Q_o & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & Q_o & \mathbf{0} \end{pmatrix}, \tag{11}$$

where $\mathbf{0}$ and Q_o are both $N \times N$ matrices. Matrix Q_o is:

$$Q_{o(N\times N)} = \begin{pmatrix} 1 & 0 & 0 & \dots & 0 & 0 \\ -1 & 1 & 0 & \dots & 0 & 0 \\ 0 & -1 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 1 & 0 \\ 0 & 0 & 0 & \dots & -1 & 1 \end{pmatrix}. \tag{12}$$

Considering both states x and holding controls u are unknown variables, we transform Eq. (8) to:

$$[I - P, -Q] \begin{bmatrix} x \\ u \end{bmatrix} = f, \tag{13}$$

where $[x; u]$ is a vector of variables. Eq. (13) is the physics dynamic constraint of the model. Restricted by practical physics rules, the problem has another three constraints as follows.

Hold Constraint: in each cell k at time t, the number of aircraft to be held is less than the current aircraft counts:

$$u \le x. \tag{14}$$

(14) is incorporated into Eq. (13) in an inequality form as the *Dynamics Constraint* for a single path:

$$\begin{bmatrix} I-P & -Q \\ P-I & Q \\ -I & I \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix} \le \begin{bmatrix} f \\ -f \\ \mathbf{0} \end{bmatrix}. \tag{15}$$

Non-negative Constraint: x and u should be non-negative:

$$x \ge 0, u \ge 0. \tag{16}$$

Integral Constraint: x and u should be integer vectors:

$$x, u \in \mathbb{I}^{NT}, \tag{17}$$

where $\mathbb{I}^{NT}$ is the integer vector domain of dimension NT.

2) Decoupled Sector Level Model: For the dynamics constraint in Eq. (15) of each path i with N_i cells, we denote the $3N_iT \times 2N_iT$ matrix as A_i, the $2N_iT$ vector consisting of x and u as x_i, and the $3N_iT$ vector on right side as f_i. Thus we obtain the decoupled all-path dynamics constraints by:

$$\begin{bmatrix} A_1 & & & \\ & A_2 & & \\ & & \ddots & \\ & & & A_M \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_M \end{bmatrix} \le \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_M \end{bmatrix} \tag{18}$$

where the matrix size is $(3\sum_{i=1}^{M} N_iT) \times (2\sum_{i=1}^{M} N_iT)$, the x vector has a length $2\sum_{i=1}^{M} N_iT$, and the length of f is $3\sum_{i=1}^{M} N_iT$. Eq. (18) describes the internal dynamics constraints for each path. All paths are decoupled.

3) Coupled Network Level Model: In practical air traffic network, several paths usually pass through a same sector with a count constraint. To be more precise, air traffic controllers even set different count constraints to one sector at different time periods, e.g. higher count during daytime and lower at night. The *Sector Count Constraint* is given by:

$$M \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_M \end{bmatrix} \leq \begin{bmatrix} \nu_1 \\ \nu_2 \\ \vdots \\ \nu_S \end{bmatrix} \tag{19}$$

where

$$\nu_j = [\nu_j(0); \nu_j(1); ...; \nu_j(T-1)]. \tag{20}$$

$\nu_j(t)$ is the sector count capacity for jth sector at time period t. $\nu = [\nu_1; \nu_2; ...; \nu_S]$ is $TS \times 1$ and x has a length $2\sum_{i=1}^{M} N_i T$. Matrix M has a dimension $TS \times 2\sum_{i=1}^{M} N_i T$, mapping aircraft counts from paths to sectors.

M consists of blocks like M_{ji} mapping aircraft counts from path i to sector j, explained as follows.

$$M = \begin{bmatrix} M_{11} & M_{12} & \dots & M_{1M} \\ M_{21} & M_{22} & \dots & M_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ M_{S1} & M_{S2} & \dots & M_{SM} \end{bmatrix}, \tag{21}$$

where the structure of block M_{ji} is:

$$M_{ji} = \left[\begin{array}{cccc|c} s'_{ji} & \mathbf{0}'_\alpha & \dots & \mathbf{0}'_\alpha & \\ \mathbf{0}'_\alpha & s'_{ji} & \dots & \mathbf{0}'_\alpha & \\ \vdots & \vdots & \ddots & \vdots & \mathbf{0}_\beta \\ \mathbf{0}'_\alpha & \mathbf{0}'_\alpha & \dots & s'_{ji} & \end{array}\right] := [M^l_{ji} | \mathbf{0}_\beta]. \tag{22}$$

Inside M_{ji}, for each path i with N_i cells, we use a vector s_{ji} of length N_i to denote which cells on path i lie in sector j:

$$s_{ji}(k) = \begin{cases} 1, & \text{if the } k\text{th cell of path } i \text{ is in sector } j; \\ 0, & \text{otherwise.} \end{cases} \tag{23}$$

while $\mathbf{0}'_\alpha$ is a zero row vector and $\mathbf{0}_\beta$ is a $T \times N_i T$ all-zero matrix. The matrix M^l_{ji} to the left of $\mathbf{0}_\beta$ is also $T \times N_i T$.

The diagonal blocks consist of the same row vector s'_{ji} because the relationship that cell k in path i belongs to sector j does not change with time.

We define M^l_i as follow:

$$M^l_i = \begin{bmatrix} M^l_{1i} \\ M^l_{2i} \\ \vdots \\ M^l_{Si} \end{bmatrix}. \tag{24}$$

Another feature inside matrix M is that the sum of each column in matrix M^l_i is 1, because in each time t, the kth cell of path i can only belong to one sector.

4) Integer Programming Formulation: We formulate the optimization problem as follows. First we denote

$$A = \left[\begin{array}{cccc} A_1 & & & \\ & A_2 & & \\ & & \ddots & \\ & & & A_M \\ \hline M_{11} & M_{12} & \dots & M_{1M} \\ M_{21} & M_{22} & \dots & M_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ M_{S1} & M_{S2} & \dots & M_{SM} \end{array}\right], b = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_M \\ \nu_1 \\ \nu_2 \\ \vdots \\ \nu_S \end{bmatrix}, \tag{25}$$

and

$$c = [c_1; \mathbf{0}_1; c_2; \mathbf{0}_2; ...; c_M; \mathbf{0}_M], \tag{26}$$

where c_i is all-one and $\mathbf{0}_i$ is all-zero. c and x are both vectors of length $2\sum_{i=1}^{M} N_i T$.

From (16) and (17), we know x is required to be non-negative and integral. According to the physics rules of (2), (3) and (20), vector b is also integral.

In summary, the original problem is formulated as an integer programming as follow:

$$\begin{array}{rl} & \min c'x \\ \text{s.t.} & Ax \leq b, \\ & x \geq 0, \text{and } x \in \mathbb{I}^{2T\sum_{i=1}^{M} N_i} \end{array} \tag{27}$$

B. Standard Linear Programming Form and Total Unimodularity

Solving the integer programming in formulation (27) is extremely time consuming and sometimes impossible for a large scale problem in air traffic management. The authors in [5] relaxed (27) to a linear programming to achieve better computational efficiency. It can be written in the *Standard Linear Programming Form* [9]:

$$\begin{array}{rl} & \min c'x \\ \text{s.t.} & Ax \leq b, \\ & x \geq 0 \end{array} \tag{28}$$

which in general results in fractional solutions [5].

1) Total Unimodularity and Integral Optimum:

Theorem 1: If A is totally unimodular and the problem (28) is feasible, there exists at least one integral optimum for formulation (28), which can be found by the simplex method.

Proof: According to Hoffman and Kruskal's theorem in [10], if A in formulation (28) is *Totally Unimodular* with the fact that vector b is integral, the feasible polyhedron $\{x | Ax \leq b, x \geq 0\}$ defined by constraints in (28) is integral. Recall that the simplex method generates its optimal solution by pivoting from one extreme point to an adjacent one around the feasible polyhedron and the extreme points are the vertice of the feasible polyhedron, the simplex method always generates an integral optimal solution. ■

TABLE I: Elementary Row (Column) Operations

1. exchanging two rows (columns);
2. multiplying a row (column) by -1;
3. adding a row (column) to another row (column).

It is evident that when the optimal solution to (28) is integral, the solution is also an optimal solution to (27). So the key point is to prove A is totally unimodular.

Lemma 1: If matrix A is full row (column) rank, the total unimodularity of A is preserved under three elementary row (column) operations listed in Table I.

Lemma 1 is observed by combining both Theorem 19.5 and (43)(ii) in [10].

2) Total Unimodularity of Matrix A:

Theorem 2: A in (28) is totally unimodular.

Proof:

We start with performing elementary column operations inside each blocked column of matrix A as shown in (25). Through a series of elementary column operations, we have transformed the matrix A in (25) into (29).

$$\left[\begin{array}{ccccccc} I & \mathbf{0} & & & & & \\ -I & \mathbf{0} & & & & & \\ \mathbf{0} & I & & & & & \\ & & I & \mathbf{0} & & & \\ & & -I & \mathbf{0} & & & \\ & & \mathbf{0} & I & & & \\ & & & & \ddots & & \\ & & & & & I & \mathbf{0} \\ & & & & & -I & \mathbf{0} \\ & & & & & \mathbf{0} & I \\ \hline L_{11} & R_{11} & L_{12} & R_{12} & \dots & L_{1M} & R_{1M} \\ L_{21} & R_{21} & L_{22} & R_{22} & \dots & L_{2M} & R_{2M} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ L_{S1} & R_{S1} & L_{S2} & R_{S2} & \dots & L_{SM} & R_{SM} \end{array}\right]. \tag{29}$$

Therefore, the upper part of (29) tells that the matrix (29) after the elementary column operations is full column rank. We know that the elementary column operations do not change the column rank of a matrix, so the matrix A before these operations is also full column rank. Thus according to Lemma 1, the elementary column operations we have performed can preserve the total unimodularity of matrix A. The problem becomes to prove (29) is totally unimodular.

Moreover, based on (43)(v) of [10], if the lower part of (29) is totally unimodular, then the whole matrix (29) is also totally unimodular. Now we need to show (30) is totally unimodular.

$$\begin{bmatrix} L_{11} & R_{11} & L_{12} & R_{12} & \dots & L_{1M} & R_{1M} \\ L_{21} & R_{21} & L_{22} & R_{22} & \dots & L_{2M} & R_{2M} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ L_{S1} & R_{S1} & L_{S2} & R_{S2} & \dots & L_{SM} & R_{SM} \end{bmatrix}, \tag{30}$$

where L_{ji} is a lower triangle blocked matrix with every non-zero block as s'_{ji} and the non-zero block t'_{ji} fills the lower triangle positions of matrix R_{ji} except the main diagonal.

$$L_{ji} = \begin{bmatrix} s'_{ji} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ s'_{ji} & s'_{ji} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ s'_{ji} & s'_{ji} & s'_{ji} & \dots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ s'_{ji} & s'_{ji} & s'_{ji} & \dots & s'_{ji} & \mathbf{0} \\ s'_{ji} & s'_{ji} & s'_{ji} & \dots & s'_{ji} & s'_{ji} \end{bmatrix}, \tag{31}$$

$$R_{ji} = \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ t'_{ji} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ t'_{ji} & t'_{ji} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ t'_{ji} & t'_{ji} & t'_{ji} & \dots & \mathbf{0} & \mathbf{0} \\ t'_{ji} & t'_{ji} & t'_{ji} & \dots & t'_{ji} & \mathbf{0} \end{bmatrix}. \tag{32}$$

In (30) we assume that every sector contains at least one cell from some certain path. If there is a sector containing no cells, the corresponding blocked row can be deleted by (43)(v) in [10]. In this case we actually do not need to include this sector in our model.

We perform elementary row operations to (30) and get (33):

$$\begin{bmatrix} \widetilde{L}_{11} & \widetilde{R}_{11} & \widetilde{L}_{12} & \widetilde{R}_{12} & \dots & \widetilde{L}_{1M} & \widetilde{R}_{1M} \\ \widetilde{L}_{21} & \widetilde{R}_{21} & \widetilde{L}_{22} & \widetilde{R}_{22} & \dots & \widetilde{L}_{2M} & \widetilde{R}_{2M} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \widetilde{L}_{S1} & \widetilde{R}_{S1} & \widetilde{L}_{S2} & \widetilde{R}_{S2} & \dots & \widetilde{L}_{SM} & \widetilde{R}_{SM} \end{bmatrix}, \tag{33}$$

in which matrices $\widetilde{L}_{ji}$ and $\widetilde{R}_{ji}$ are:

$$\widetilde{L}_{ji} = \begin{bmatrix} s'_{ji} & & & & & \\ & s'_{ji} & & & & \\ & & s'_{ji} & & & \\ & & & \ddots & & \\ & & & & s'_{ji} & \\ & & & & & s'_{ji} \end{bmatrix}, \tag{34}$$

$$\widetilde{R}_{ji} = \begin{bmatrix} \mathbf{0} & & & & & \\ t'_{ji} & \mathbf{0} & & & & \\ & t'_{ji} & \mathbf{0} & & & \\ & & \ddots & \ddots & & \\ & & & t'_{ji} & \mathbf{0} & \\ & & & & t'_{ji} & \mathbf{0} \end{bmatrix}. \tag{35}$$

As we have assumed previously there must be at least one 1 appearing at certain cell of s'_{ji} in every row of a certain $\widetilde{L}_{ji}$ of each blocked row. As we have shown at the end of III-A3, if $\widetilde{L}_i$ is defined as:

$$\widetilde{L}_i = \begin{bmatrix} \widetilde{L}_{1i} \\ \widetilde{L}_{2i} \\ \vdots \\ \widetilde{L}_{Si} \end{bmatrix}. \tag{36}$$

the sum of each column in $\widetilde{L}_i$ is 1, which means the 1 appearing at a certain cell of s'_{ji} can not be represented by other rows, in other words, every row is independent. Since (33) is full row rank, the elementary row operations we have performed also preserve the total unimodularity. Our next concern is whether (33) is totally unimodular.

Since the column sum of every $\widetilde{L}_i$ is 1, according to (43)(v) in [10], the problem is equivalent to proving (37) is totally unimodular.

$$\begin{bmatrix} \widetilde{R}_{11} & \widetilde{R}_{12} & \dots & \widetilde{R}_{1M} \\ \widetilde{R}_{21} & \widetilde{R}_{22} & \dots & \widetilde{R}_{2M} \\ \vdots & \vdots & \vdots & \vdots \\ \widetilde{R}_{S1} & \widetilde{R}_{S2} & \dots & \widetilde{R}_{SM} \end{bmatrix}, \tag{37}$$

In our model we notice that a path consists at most one link in a sector. Thus there can be only one consecutive 1's series in s_{ji}. In fact, even if there are multiple consecutive 1's series in s_{ji}, our proof still holds.

t'_{ji} in (32) and (35) records what kind of 0-1 changes at which cells in corresponding s'_{ji}. For example, if $s'_{ji} = [0,0,1,1,1,0]$, the corresponding $t'_{ji} = [0,-1,0,0,1,0]$. -1 at the 2nd cell of t'_{ji} means a change from 0 to 1 between the 2nd cell and the 3rd cell in s'_{ji}. 1 at the 5th cell of t'_{ji} represents a change from 1 to 0 between the 5th cell and the 6th cell in s'_{ji}. Note that there is always a hiden 0 to the right of s'_{ji}. That means if the last cell of s'_{ji} is 1, it is considered as a change from 1 to 0. So there will be a 1 at the last cell in the corresponding t'_{ji}.

Consider the ith blocked column of matrix in (37), it describes all the 0-1 changing information of N_i cells of the ith path in all S sectors. When there is a -1 at a certain cell of a certain t'_{ji} in a certain row, there must be one and only one 1 at the same cell in a different row. Because when there is a change from 0 to 1 at a certain cell of s_{ji}, which means the path starts to enter sector j at this cell, there must be one and only one change from 1 to 0 in another s_{ki}, which means the ith path leaves the previous sector k right before sector j at this cell. It is also true vice versa. Based on (43)(v) in [10], after deleting all-zero columns in (37) at which cells no 0-1 change happens and the last column of each blocked column in (37), where there is only one non-zero item 1 in a certain row, the modified matrix (37) only contains the columns having exactly one 1/-1 pair.

According to (18) of [10], the matrix whose each column contains exactly one 1 and exactly one -1 is totally unimodular.

In summary, since all the operations we have performed can preserve the total unimodularity of a matrix, and the final transformed matrix has been proved to be totall unimodular, the original matrix A is totally unimodular.

■

Since matrix A is totally unimodular and vector b is integral, there must exist an integral optimum both for LP relaxation (28) and for the integer programming in (27) when (28) is feasible.

IV. Why Interior-Point Method Has Fractional Solution?

The major reason is because of the optimal degeneracy of the LP relaxation. The optimal degeneracy means that there are multiple optimal extreme point solutions in this problem. These optimal extreme points will form an *Optimal Polyhedron* which is the subset of the *Feasible Polyhedron* defined by the constraints.

Generally P_{fs} is the feasible polyhedron defined by constraints while the optimal polyhedron P_{opt} is resulted by degeneracy of multiple optimal extreme points. Any solution in P_{opt} is a linear combination of the optimal extreme point solutions and can be written as $\sum_i^{|opt|} \rho_i \cdot opt_i$, with $\sum_i^{|opt|} \rho_i = 1$, where $|opt|$ is the number of optimal extreme point solutions.

Although every optimal vertex solution opt_i is integral because of total unimodularity, the linear combination of them can not be guaranteed integral. There are fractional optimal solutions inside P_{opt}. Since the interior-point method starts inside P_{fs} and walks toward the edges of P_{fs} instead of the vertices, an inner point of the subset P_{opt} is usually achieved. That is why the interior-point method gives out fractional optimum in [5].

V. A Revised Dantzig-Wolfe Decomposition Paradigm on Large Scale Study

We decide to choose simplex method for large scale cases of TFM problems. In order to speed up it by taking advantage of A's sparsity and *Block-Angular Structure* [9], we exploit *Dantzig-Wolfe Decomposition* (DWD) [11]. Based on the simplex method [9], the DWD guarantees an integral optimum.

A. Rearrangement for Dantzig-Wolfe Decomposition

Formulation (25) is rearranged into a required form for DWD. We group blocks $M_{1i}, M_{2i}, ..., M_{Si}$ in column i into a single block called M_i for $i = 1, 2, ..., M$, and switch the positions of M_i's and the dynamics constraints A_i's. The new constraint matrix A_{DW} and vector b_{DW} are:

$$A_{\text{DW}} = \begin{bmatrix} M_1 & M_2 & \dots & M_M \\ A_1 & & & \\ & A_2 & & \\ & & \ddots & \\ & & & A_M \end{bmatrix}, b_{\text{DW}} = \begin{bmatrix} \nu \\ f_1 \\ f_2 \\ \vdots \\ f_M \end{bmatrix}, \tag{38}$$

where the grouped sector counts capacity is $\nu = [\nu_1; \nu_2; ...; \nu_S]$. The cost vector c_{DW} in (26) can be written as:

$$c_{\text{DW}} = [c_{\text{DW}_1}; c_{\text{DW}_2}; ...; c_{\text{DW}_M}], \tag{39}$$

where $c_{\text{DW}_i} = [c_i; \mathbf{0}_i]$, with c_i and $\mathbf{0}_i$ from (26).

This problem can be solved by DWD, which transforms the original problem into the *Master Problem* [9].

B. Classical Dantzig-Wolfe Decomposition Algorithm

For the ith subproblem as below:

$$\begin{array}{rl} & \min c'_{\text{DW}_i} x_i \\ \text{s.t.} & A_i x_i \le f_i \end{array} \tag{40}$$

there are V_i extreme points $x_i^{(j)}$, $j = 1, 2, ..., V_i$. Also we define $P_{ij} = M_i x_i^{(j)}$ and $c_{ij} = c'_{\text{DW}_i} x_i^{(j)}$.

The c_{MP}, A_{MP}, b_{MP} of the master problem [9] are:

$$c_{\text{MP}} = [c_{11} ... c_{1V_1} c_{21} ... c_{2V_2} ... c_{M1} ... c_{MV_M}]',$$

$$A_{\text{MP}} = \begin{bmatrix} P_1 & P_2 & \dots & P_M \\ \mathbf{1}'_{\gamma 1} & & & \\ & \mathbf{1}'_{\gamma 2} & & \\ & & \ddots & \\ & & & \mathbf{1}'_{\gamma M} \end{bmatrix}, b_{\text{MP}} = \begin{bmatrix} \nu \\ 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}, \tag{41}$$

where $P_i = [P_{i1}, ..., P_{iV_i}]$ and $\mathbf{1}'_{\gamma i} = [1, ..., 1]$ with V_i all 1's.

A revised simplex method is used in performing the calculations in DWD [11]. In the revised simplex method, there is a vector of "prices" $[\pi', \tilde{\pi}']$, where π is of length TS and $\tilde{\pi}$ is of M. Each item of the price vector is associated with one constraint in formulation (41).

Starting from an initial basis, an iterative DWD process begins, where the master problem transfers the updated price vector to the original subproblems while the subproblems provide the entering base which has the minimum negative reduced cost. At the same time, a column will be deleted by simplex rule. Since we have a new basis, we can update the price vector and transfer it to subproblems again. The iteration will be terminated until there is no negative reduced cost.

C. Degeneracy-Aware DW Decomposition Algorithm

Although DWD is a general solution for block angular structure problems, it can not be applied in this problem because two degeneracies exist. They are *Feasible Degeneracy* and *Optimal Degeneracy*. A new algorithm, which is called Degeneracy-Aware Dantzig-Wolfe Decomposition (DADWD), is developed in this paper to deal with the two degeneracies.

1) Feasible Degeneracy: Back in Eq. (18), the subproblem variable x_i has length $2N_iT$, where some variables are dependent on others, e.g. the first N_iT states of x are determined by the last N_iT holding controls u through the dynamics in (4). The number of independent variables is then reduced to N_iT. Moreover, the first N_i states of x at time 0 are the initial conditions, which do not contribute to variable space rank. The last N_i holding controls at time $T-1$ do not affect the solution because they only determine the states x out of our planning horizon. Therefore, in a subproblem (40), the variable space rank of x_i is compressed into $N_i(T-1)$.

For each subproblem we treat states x as variables and holding controls u as correlated variables. Since the first N_i states at time 0 are predetermined by initial condition, a series of truncations are needed to delete the rows or columns in A_i, M_i, f_i and ν associated with the first N_i states of x. The truncated $\hat{A}_i$, $\hat{M}_i$, $\hat{f}_i$ and $\hat{\nu}$ with lower dimensions are input to the DADWD algorithm .

After the truncations, the number of basis for the master problem is $S(T-1)+M$. The *Initial Basis Generation* (IBG) is activated. For each path, hold the count of the first cell inside sector i at time t and let other flows hop to their next cells. That will give out $S(T-1)$ basis because the holding controls at time $T-1$ have no effect. The remaining M basis are filled by columns of holding control at time $T-1$ cascaded with e_i, which is a M length vector where the ith element is 1 and others are 0.

2) Optimal Degeneracy: We find multiple optima in Sec. IV. Multiple optima inside each subproblem make it difficult to apply the classical DWD.

Multiple optima are discussed in the classical simplex method. However, most algorithms enumerating multiple optima are NP-hard [12]. Therefore a conjugate gradient projection method is implemented to find all extreme point optima for each subproblem. The algorithm of [13] is referred as *Feasible Direction Method* (FDM) in this paper.

In summary, the final DADWD is given in Algorithm 1.

Algorithm 1 Degeneracy-Aware DW Decomposition

```
 1: Initialization and Truncation
 2: Run Initial Basis Generation
 3: while TRUE do
 4:     Solve the Master Problem
 5:     Update the price vector based on current basis
 6:     for $i = 1$ to $M$ do
 7:         Plug $\pi$ and $\tilde{\pi}$ into each subproblem $i$
 8:         Find multiple optima of subproblem $i$ by FDM [13]
 9:         if the reduced cost of the optimal solutions $< 0$ then
10:             Add the corresponding columns into basis
11:         end if
12:     end for
13:     if no negative reduced cost appeared then
14:         Stop
15:     end if
16: end while
```

VI. Conclusion

In this paper, CTM(L) is introduced and an integer programming optimization problem is formulated. We prove that there exists an integral optimal solution for the associated LP relaxation because of its total unimodularity and this solution is also optimal for the integer programming. We demonstrate that the simplex method guarantees the integral optimum and propose the revised simplex based Degeneracy-Aware Dantzig-Wolfe Decomposition algorithm which takes advantage of matrix A's special block structure to speed up the computation while handling the feasible degeneracy and optimal degeneracy together. The new algorithm can be considered as a potential framework for parallel computation in the practical large scale models.

References

[1] A. Odoni, "The flow management problem in air traffic control," in *Flow Control of Congested Networks*. Berlin, Germany: Springer Verlag, 1987.

[2] M. Helme, "Reducing air traffic delay in a space-time network," in *IEEE International Conference on Systems, Man and Cybernetics*, vol. 1, 1992, pp. 236–242.

[3] K. Lindsay, E. Boyd, and R. Burlingame, "Traffic flow management modeling with the time assignment model," *Air Traffic Control Quarterly*, vol. 1, no. 3, pp. 255–276, 1993.

[4] B. Sridhar, G. Chatterji, S. Grabbe, and K. Sheth, "Integration of traffic flow management decisions," in *AIAA Conference on Guidance, Navigation, and Control Conference and Exhibit*, Monterey, CA, August 2002.

[5] D. Sun and A. Bayen, "Multicommodity eulerian-lagrangian large-capacity cell transmission model for en route traffic," *AIAA Journal of Guidance, Control, and Dynamics*, vol. 31, no. 2, pp. 616–628, 2008.

[6] P. Menon, G. Sweriduk, and K. Bilimoria, "A new approach to modeling, analysis and control of air traffic flow," in *Proceedings of AIAA Guidance, Navigation and Control Conference*, Monterey, CA, 2002.

[7] A. Bayen, R. Raffard, and C. Tomlin, "Adjoint-based control of a new eulerian network model of air traffic flow," *IEEE Trans. Control Syst. Technol.*, vol. 14, no. 5, pp. 804–818, 2006.

[8] C. Daganzo, "The cell transmission model: a dynamic representation of highway traffic consistent with the hydrodynamic theory," *Transportation Research*, vol. 28B, no. 4, pp. 269–287, 1994.

[9] V. Chvatal, *Linear Programming*. W. H. Freeman, 1983.

[10] A. Schrijver, *Theory of Linear and Integer Programming*. Wiley, 1998.

[11] G. Dantzig and P. Wolfe, "The decomposition algorithm for linear programs," *Econometrica*, vol. 29, no. 4, pp. 767–778, 1961.

[12] H. Arsham, "Tools for modeling validation process: The dark side of lp," Educational Multimedia, Tech. Rep., 1994.

[13] S. Tantawy, "Using feasible direction to find all alternative extreme opti al points for linear programming problem," *Journal of Mathematics and Statistics*, vol. 3, no. 3, pp. 109–111, 2007.